Group Art Unit 2162
Docket No: SVL920030088US1

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPEAL BRIEF – 37 C.F.R § 1.192

U.S. Patent Application 10/605,448 entitled:

"*Extensible Decimal Identification System for Ordered Nodes*"

Real Party in Interest: International Business Machines Corporation

Related Appeals and Interferences:

None

Status of Claims:

Claims 1-22 were previously canceled.

Claims 23-44 are pending.

Claims 23-44 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over O'Neil et al. (U.S. Patent 6,889,226), hereafter O'Neil, in view of Rizzo et al. (U.S. Published Application 2004/0068500), hereafter Rizzo).

Claims 23-44.

Claims 23-44 are hereby appealed.

Status of Amendments:

No amendments after the Final Rejection dated 1/23/2008, have been filed.

Summary of Claimed Subject Matter:

(NOTE: All citations are made from the corresponding US Pre-Grant Publication **2006/0173927**.)

The present invention according to independent **claim 23** provides for a robust computer-based method for updating a computer-stored hierarchical structure of nodes via a node identification technique, said nodes of said hierarchical structure stored as encoded values in a computer storage, said update retaining properties and parent/child relationships of said

hierarchical structure without renumbering existing node ID values associated with said hierarchical structure, said method comprising the steps of: (a) receiving an instruction to insert a new node at an insertion point in said computer-stored hierarchical structure **(see at least paragraphs [0022] and [0024] of the application-as-filed)**; (b) identifying one of, or a combination of the following: a left node ID value closest to the left of said insertion point or a closest right node ID value closest to the right of said insertion point **(see at least figures 3 and 4 and paragraphs [0022] and [0024] of the application-as-filed)**; (c) calculating a new ID value based upon node ID value(s) identified in (b), said calculated value greater than ID values of nodes to the left of said insertion point and less than ID values of nodes to the right of said insertion point, said new ID value based upon a low/high key value, said high key value representing a highest encodable value and said low key value representing a lowest encodable value **(see at least figures 3 and 4 and paragraphs [0022], [0023],[0024], [0025], and [0026] of the application-as-filed)**; and (d) encoding said calculated new ID value and updating said computer storage storing said nodes of said hierarchical structure with said encoded value, wherein order, node ID values, and relationships between parent, child, and siblings in said hierarchical structure of nodes stored in said storage remain unchanged with said insertion of new node **(see at least figures 3 and 4 and paragraphs [0022] and [0024] of the application-as-filed)**.

The present invention according to independent **claim 31** provides for an article of manufacture comprising a computer usable medium having computer readable program code embodied therein which updates a computer-stored hierarchical structure of nodes via a node identification technique, said nodes of said hierarchical structure stored as encoded values in a

computer storage, said update retaining properties and parent/child relationships of said hierarchical structure without renumbering existing node ID values associated with said hierarchical structure, said medium comprising: (a) computer readable program code aiding in receiving an instruction to insert a new node at an insertion point in said computer-stored hierarchical structure **(see at least paragraphs [0022], [0024], [0035], and [0036] of the application-as-filed)**; (b) computer readable program code identifying one of, or a combination of the following: a left node ID value closest to the left of said insertion point or a closest right node ID value closest to the right of said insertion point **(see at least figures 3 and 4 and paragraphs [0022], [0024], [0035], and [0036] of the application-as-filed)**; (c) computer readable program code calculating a new ID value based upon node ID value(s) identified in (b), said calculated value greater than ID values of nodes to the left of said insertion point and less than ID values of nodes to the right of said insertion point, said new ID value based upon a low/high key value, said high key value representing a highest encodable value and said low key value representing a lowest encodable value **(see at least figures 3 and 4 and paragraphs [0022], [0023],[0024], [0025], [0026], [0035], and [0036]of the application-as-filed)**; and (d) computer readable program code encoding said calculated new ID value and updating said computer storage with said encoded value, wherein order, node ID values, and relationships between parent, child, and siblings in said hierarchical structure of nodes stored in said storage remain unchanged with said insertion of new node **(see at least figures 3 and 4 and paragraph [0022], [0024], [0035], and [0036] of the application-as-filed)**.

The present invention according to independent **claim 39** provides a computer-based method for updating a computer-stored hierarchical structure of nodes without renumbering

existing node ID values associated with said hierarchical structure, said nodes of said hierarchical structure stored as binary encoded values in a computer storage, said method comprising the steps of: (a) receiving an instruction to insert a new node at an insertion point in said computer-stored hierarchical structure **(see at least paragraphs [0022] and [0024] of the application-as-filed)**; (b) identifying one of, or a combination of the following: a left node ID value closest to the left of said insertion point or a closest right node ID value closest to the right of said insertion point **(see at least figures 3 and 4 and paragraphs [0022] and [0024] of the application-as-filed)**; (c) calculating a new ID value for node to be inserted based upon a low key value 0 or a high key value x, said high key value representing a highest binary encodable value and said low key value representing a lowest binary encodable value, said calculation performed via one of the follows ways: concatenating said left node ID value with one or more high key values and a positive value or concatenating said left node ID value with one or more low key values and a positive value **(see at least figures 3 and 4 and paragraphs [0022], [0023],[0024], [0025], and [0026] of the application-as-filed)**; and (d) encoding, via binary encoding, said calculated new ID value and updating said computer storage with said binary encoded value, wherein order, node ID values, and relationships between parent, child, and siblings in said hierarchical structure of nodes stored in said storage remain unchanged with said insertion of new node **(see at least figures 3 and 4 and paragraphs [0022] and [0024] of the application-as-filed)**.

Grounds of Rejection to be Reviewed on Appeal:

Claims 23-44 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over O'Neil et al. (U.S. Patent 6,889,226), hereafter O'Neil, in view of Rizzo et al. (U.S. Published Application 2004/0068500), hereafter Rizzo). Was a proper rejection made under 35 U.S. C. § 103(a) using existing USPTO guidelines?

ARGUMENT:

Claims 23-44 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over O'Neil et al. (U.S. Patent 6,889,226), hereafter O'Neil, in view of Rizzo et al. (U.S. Published Application 2004/0068500), hereafter Rizzo). Was a proper rejection issued under 35 U.S.C. § 103(a)?

To be properly rejected under 35 U.S.C. §103(a), the prior art reference (or references when combined) must teach or suggest all the claim limitations. Applicants respectfully assert that the combination of O'Neil and Rizzo fails to render obvious many of the features of Applicants pending claimed.

O'Neil teaches a technique for representing the structure of hierarchically-organized data in a non-hierarchical data structure, such as a relation, wherein the hierarchically-organized data is represented as a tree, and each node in the tree is assigned a position identifier that represents both the depth level of the node within the hierarchy, and its ancestor/descendant relationship to other nodes.

Rizzo teaches a data sorting apparatus comprising a storage sorter that sorts a data set according to a defined criteria and a query mechanism that receives intermediate sorted data values from the storage sorter and compares the intermediate sorted data values to at least one key value.

Applicants independent claim 23, by stark contrast, teaches a robust computer-based

method for updating a computer-stored hierarchical structure of nodes via a node identification technique, wherein **the nodes of the are stored as encoded values in a computer storage**, and the method comprising the steps of: (a) receiving an instruction to insert a new node at an insertion point in said computer-stored hierarchical structure; (b) identifying one of, or a combination of the following: a left node ID value closest to the left of said insertion point or a closest right node ID value closest to the right of said insertion point; (c) calculating a new ID value based upon node ID value(s) identified in (b), said calculated value greater than ID values of nodes to the left of said insertion point and less than ID values of nodes to the right of said insertion point, wherein **the new ID value based upon a low/high key value, said high key value representing a highest encodable value (e.g., 1111) and said low key value representing a lowest encodable value (e.g., 0000)**; and (d) **encoding said calculated new ID value and updating said computer storage storing said nodes of said hierarchical structure with said encoded value, wherein order, node ID values, and relationships between parent, child, and siblings in said hierarchical structure of nodes stored in said storage remain unchanged with said insertion of new node**.

The present invention provides for an extensible identification system for nodes in a hierarchy, wherein each node is assigned a concatenation of decimal based values. The identification value uniquely identifies the node, provides an order for the node, and identifies its parent, child, and sibling relationships with other nodes. Also, the IDs assigned can be encoded (via for e.g., binary encoding) to be byte comparable. Furthermore, the IDs assigned to nodes need not be modified when changes (adding/deleting a child node or a subtree of nodes) are made in the hierarchy. Additionally, in the event of such a change, the order and relationships

between the parent, child, and sibling nodes are retained.

The present invention provides a way for assigning IDs to nodes in a hierarchy and provides many advantages, some of which include: (a) the IDs provide a way of ordering nodes in a hierarchy; (b) the IDs describe a node's parent, child, and sibling relationships; (c) the IDs can be encoded such that they are byte comparable; (d) the IDs can be assigned to newly inserted nodes, anywhere in the hierarchy, and still maintain these properties; and (e) the IDs, once assigned, do not have to change even with changes to the hierarchy.

The presently claimed invention can be distinguished from O'Neil because the claimed invention uses the notion of a positive infinity number 'x' (representing the highest encodable value such as 1111) and negative infinity number '0' (representing the lowest encodable value such as 0000) to define the boundary of subtrees (see, for example, Table 3 of the application-as-filed). For example, to insert between 1.1 and 1.2, we use the number 1.1.x.1. The x (positive infinity) is higher than any value that can represent any node within the subtree under 1.1. So under 1.1, the first child could be 1.1.1, second child is 1.1.2, third 1.1.3 and so on. But a child of 1.1 can never be equal to or greater than 1.1.x because x is higher than any value. Because of this, the range $1.1 < a < 1.1.x$ can be used to define the nodes within the subtree of 1.1. The same argument applies to '0', where 0 represents negative infinity (which is used to go in the opposite direction).

O'Neil et al.'s Figures 5 and 6 show how data can be inserted (or "careted") into a hierarchical data structure. However, it should be noted that O'Neil's structure is restrictive in

the fact that only odd numbers are used as position numbers for nodes. As an example, O'Neil's figures 5 and 6 is reproduced below:

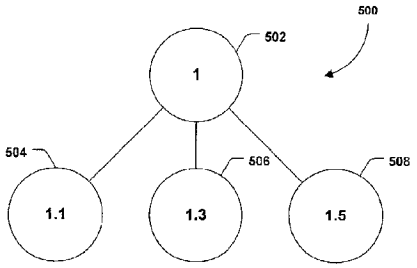

FIG. 5

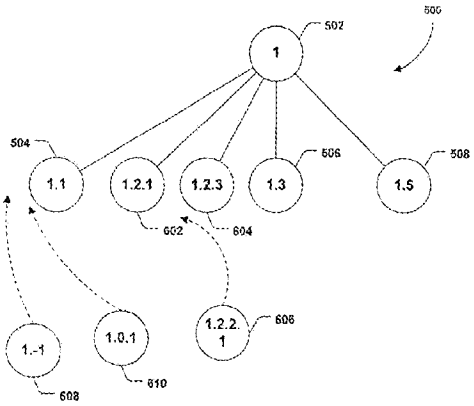

FIG. 6

It can be seen from the Figures 5 and 6 that nodes are numbers with odd numbers (1.1, 1.2.1, 1.2.3, 1.3, 1.5, etc.). , as by O'Neil's own admission (in column 8, lines 36+) "odd numbers are used in the position numbers for nodes 502-508; in a preferred embodiment, even numbers are explicitly omitted from the numbering scheme." Further, the only manner in which O'Neil contemplates having consecutive integers, by their own admission in column 8, lines 48-52, is when "it is not necessary to perform insertions in a manner that captures the order of a new node relative to its siblings."

By stark contrast, the claimed invention's node insertion/deletion scheme is more robust as it is NOT limited by considerations of even and odd nodes, but is rather dependent on low and high key values. For example, the Board of Patent Appeals and Interferences is hereby respectfully requested to review Figure 4 of the application-as-filed which is reproduced below:

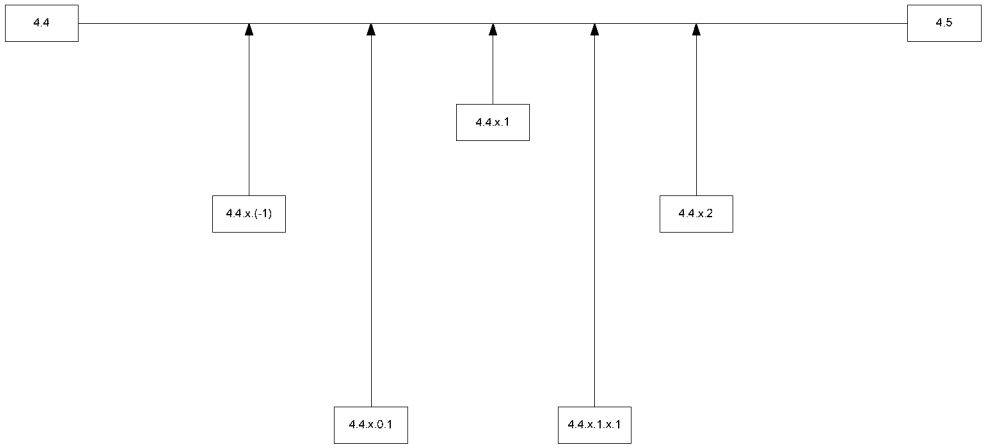

Figure 4

It can be seen from the above figure that the insertion of nodes do not rely on a pre-defined scheme that utilizes odd numbers to represent nodes, but is based upon a low/high key value. Specifically, if a new node is to be inserted between existing node 4.4 and 4.5, the present invention inserts a new node based on concatenating 'x.1' to the left node 4.4 to form new node '4.4.x.1', wherein **'x' is the high key value**. If a new node is to be inserted between node 4.1 and the newly created node '4.4.x.1', a **low key value of '0'** is used to form new node 4.4.x.0.1.

A simple review of the Figures of O'Neil's re-emphasizes Applicants' point that **O'Neil fails to disclose any insertion or deletion based on high or low key values**. Applicants' independent claim 23, by stark contrast, specifically requires the **calculation of a new ID value based on such high or low key values.** The claimed invention node insertion/deletion scheme is more robust as it is NOT limited by considerations of even and odd nodes.

It should be noted that the secondary references used by the Examiner, i.e., Rizzo, does not relate, in any manner, to the storage, updating or deleting of nodes. Further, as mentioned

above, Rizzo merely teaches a **data sorting apparatus** comprising a storage sorter that sorts a data set according to a defined criteria and a query mechanism that receives intermediate sorted data values from the storage sorter and **compares the intermediate sorted data values to at least one key value**. **The context of "key value" as used in Rizzo is entirely different than that of the present invention and that of values described in O'Neil**. A key value, as used by Rizzo, merely refers **a value that intermediate sorted data values of compared against**. **O'Neil does not have any teachings for such a value that is to be compared against for sorting purposes (O'Neil is not concerned with sorting), and even if it did, it would not be the same as the high and low key values that are used to calculated new ID values as per the teachings of the present invention**. Rizzo further states that the 'key field' associated with the 'key value' can have a range associated with it, wherein the range is from negative infinity (-INF) to positive infinity (+INF). Applicants are unsure how the Examiner is combining such a disparate teaching of a compared value associated with data sorting to Applicants' high and low key values that are used in assigning node ID values in a hierarchical structure.

Applicants therefore maintain that **there is no teaching for Rizzo's key field range to be used to represent low and high key values in node ID calculations**. Further, the Examiner has provided **no evidence** for **how the combination of O'Neil's hierarchical structure and Rizzo's data sorter would have provided a teaching for a key field range that is used to represent low and high key values in node ID calculations**. Applicants also respectfully assert that the Examiner has failed to show any **evidence** of **why such a mere range can be interpreted to represent low and high key values that can be combined with the teachings of Rizzo to calculate node values**.

The burden of combining references cannot be satisfied by simply asserting that the modification would have been "well within the ordinary skill of the art." As the CAFC stresses for a §103 rejection to stand, the Examiner is required to show **with evidence** the desirability of making the specific combination at issue. That evidence is required to counter the powerful attraction of a hindsight-based obviousness analysis. See, for example, In re Lee, 277 F.3d 1338, 1343, 61 U.S.P.Q. 2d 1430, 1433 (Fed. Cir. 2002) ("Our case law makes clear that the best defense against the subtle but powerful attraction of a hindsight-based obviousness analysis is rigorous application of the requirement for a showing of the teaching or motivation to combine prior art references"). It is respectfully submitted that this involves more than a mere bold assertion that it would be obvious to combine the cited references. With respect, the Examiner has failed to provide any evidence as to why one of ordinary skill in the art would be motivated to combine the teachings of O'Neil and Rizzo.

In re Lee requires that the record must state with particularity all the evidence and rationale on which the PTO relies for a rejection and sets out that it is necessary to explain the reasons one of ordinary skill in the art would have been motivated to select the references and to combine them to render the claimed invention obvious. Also, under Lee, the PTO must state in writing the evidence on which it bases its rejection. With respect, the Examiner's office action falls short of this requirement.

At least for the reasons set forth above, the combination of O'Neil and Rizzo cannot teach or suggest the features of independent claim 23.

Applicants also assert that O'Neil and Rizzo, either singularly or in combination, fail to teach or suggest claim 23's feature of **encoding said calculated new ID value and updating said computer storage storing said nodes of said hierarchical structure with said encoded value,** wherein **order, node ID values, and relationships between parent, child, and siblings in said hierarchical structure of nodes stored in said storage remain unchanged with said insertion of new node**.

Absent such teachings, the combination of O'Neil and Rizzo cannot teach or suggest the features of independent claim 23. Hence, Applicants respectfully contend that the combination of O'Neil and Rizzo cannot render obvious the teachings of Applicants' claim 23.

The present invention according to independent **claim 31** provides for an article of manufacture comprising a computer usable medium having computer readable program code embodied therein which updates a computer-stored hierarchical structure of nodes via a node identification technique, **said nodes of said hierarchical structure stored as encoded values in a computer storage**, said update retaining properties and parent/child relationships of said hierarchical structure without renumbering existing node ID values associated with said hierarchical structure, said medium comprising: (a) computer readable program code aiding in receiving an instruction to insert a new node at an insertion point in said computer-stored hierarchical structure; (b) computer readable program code identifying one of, or a combination of the following: a left node ID value closest to the left of said insertion point or a closest right node ID value closest to the right of said insertion point; (c) computer readable program code

calculating a new ID value based upon node ID value(s) identified in (b), said calculated value greater than ID values of nodes to the left of said insertion point and less than ID values of nodes to the right of said insertion point, said new ID value based upon a low/high key value, said high key value representing a highest encodable value and said low key value representing a lowest encodable value; and (d) computer readable program code encoding said calculated new ID value and updating said computer storage with said encoded value, wherein **order, node ID values, and relationships between parent, child, and siblings in said hierarchical structure of nodes stored in said storage remain unchanged with said insertion of new node**.

Claim 31 discloses an article of manufacture implementing computer readable program code implementing the method of independent claim 23. Therefore, the above-mentioned arguments with respect to independent claim 23 substantially apply to independent claim 31. Therefore, at least for the reasons set forth above, Applicants respectfully assert that the combination of O'Neil and Rizzo cannot render obvious the teachings of Applicants' independent claim 31.

The present invention according to independent **claim 39** provides a computer-based method for updating a computer-stored hierarchical structure of nodes without renumbering existing node ID values associated with said hierarchical structure, **said nodes of said hierarchical structure stored as binary encoded values in a computer storage**, said method comprising the steps of: (a) receiving an instruction to insert a new node at an insertion point in said computer-stored hierarchical structure; (b) identifying one of, or a combination of the

following: a left node ID value closest to the left of said insertion point or a closest right node ID value closest to the right of said insertion point; (c) **calculating a new ID value for node to be inserted based upon a low key value 0 or a high key value x, said high key value representing a highest binary encodable value and said low key value representing a lowest binary encodable value, said calculation performed via one of the follows ways: concatenating said left node ID value with one or more high key values and a positive value or concatenating said left node ID value with one or more low key values and a positive value**; and (d) **encoding, via binary encoding, said calculated new ID value and updating said computer storage with said binary encoded value, wherein order, node ID values, and relationships between parent, child, and siblings in said hierarchical structure of nodes stored in said storage remain unchanged with said insertion of new node**.

Applicants' independent claim 39 recites many similar features as Applicants independent claim 23. Therefore, the above-mentioned arguments with respect to independent claim 23 substantially apply to independent claim 31.

As seen from above, claim 39 specifically teaches the feature of calculating a new ID value for node to be inserted based upon a low key value 0 or a high key value x (the high key value representing a highest binary encodable value and the low key value representing a lowest binary encodable value), wherein the calculation is performed via one of the follows ways: concatenating said left node ID value with one or more high key values and a positive value or concatenating said left node ID value with one or more low key values and a positive value. Applicants' claim 39 specifically deals with the instance of **a node being inserted between a**

left node ID and right node ID, wherein the new node ID is calculated based upon a concatenation of the left node ID value with either a high key value(s) and a positive value or a low key value(s) with a positive value. For the examiner's argument, he/she cites the addition of node 610 prior to the left most node 1.1. However, in the Examiner's example, **there is no left node ID as the only node ID is the right node ID, which is 1.1**. With no left node ID, it would be erroneous to suggest that the addition of node 610 is in the same as the addition as taught by claim 39.

At least for the reasons set forth above, the combination of O'Neil and Rizzo cannot teach or suggest the features of independent claim 39.

The above-mentioned arguments with respect to the independent claims 23, 31, and 39 substantially apply to the dependent claims 24-30, 32-38, 40-44 as they inherit all the features of the claim from which they depend. Therefore, at least for the reasons set forth above, Applicants respectfully assert that the combination of O'Neil and Rizzo cannot render obvious the teachings of Applicants' dependent claims 24-30, 32-38, 40-44.

Therefore, Applicants respectfully submit that an improper 35 U.S.C. §103(a) rejection was issued with respect to pending claims 23-44.

SUMMARY

As has been detailed above, none of the references, cited or applied, provide for the specific claimed details of applicant's presently claimed invention, nor render them obvious. It is believed that this case is in condition for allowance and reconsideration thereof and early issuance is respectfully requested.

As this Appeal Brief has been timely filed within the set period of response, no fee for extension of time is required. However, the Commissioner is hereby authorized to charge any deficiencies in the fees provided, including extension of time, to Deposit Account No. 09-0460.

Respectfully submitted by
Applicant's Representative,

/ramraj soundararajan/

Ramraj Soundararajan
Reg. No. 53832

IP AUTHORITY, LLC
4821A Eisenhower Ave
Alexandria, VA 22304
(703) 461-7060

July 8, 2008

Claims Appendix:

23. (Previously Presented) A robust computer-based method for updating a computer-stored hierarchical structure of nodes via a node identification technique, said nodes of said hierarchical structure stored as encoded values in a computer storage, said update retaining properties and parent/child relationships of said hierarchical structure without renumbering existing node ID values associated with said hierarchical structure, said method comprising the steps of:

(a) receiving an instruction to insert a new node at an insertion point in said computer-stored hierarchical structure;

(b) identifying one of, or a combination of the following: a left node ID value closest to the left of said insertion point or a closest right node ID value closest to the right of said insertion point;

(c) calculating a new ID value based upon node ID value(s) identified in (b), said calculated value greater than ID values of nodes to the left of said insertion point and less than ID values of nodes to the right of said insertion point, said new ID value based upon a low/high key value, said high key value representing a highest encodable value and said low key value representing a lowest encodable value; and

(d) encoding said calculated new ID value and updating said computer storage storing said nodes of said hierarchical structure with said encoded value, wherein order, node ID values, and relationships between parent, child, and siblings in said hierarchical structure of nodes stored in said storage remain unchanged with said insertion of new node.

24. (Original) A computer-based method as per claim 23, wherein said new ID value is calculated via any of the following steps: concatenating said left node ID value with one or more high key values and a positive value or concatenating said left node ID value with one or more low key values and a positive value.

25. (Original) A computer-based method as per claim 23, wherein a digit in said calculated ID value has a negative value.

26. (Previously Presented) A computer-based method as per claim 23, wherein said encoding is binary encoding and said highest encodable value is 1111 and said lowest encodable value is 0000.

27. (Original) A computer-based method as per claim 23, wherein said ID values are encoded and are byte comparable.

28. (Original) A computer-based method as per claim 23, wherein said nodes are associated with a mark-up language based document.

29. (Original) A computer-based method as per claim 28, wherein said mark-up based language is XML.

30. (Original) A computer-based method as per claim 23, wherein said method is implemented in conjunction with a relational database.

31. (Previously Presented) An article of manufacture comprising a computer usable

medium having computer readable program code embodied therein which updates a computer-stored hierarchical structure of nodes via a node identification technique, said nodes of said hierarchical structure stored as encoded values in a computer storage, said update retaining properties and parent/child relationships of said hierarchical structure without renumbering existing node ID values associated with said hierarchical structure, said medium comprising:

(a) computer readable program code aiding in receiving an instruction to insert a new node at an insertion point in said computer-stored hierarchical structure;

(b) computer readable program code identifying one of, or a combination of the following: a left node ID value closest to the left of said insertion point or a closest right node ID value closest to the right of said insertion point;

(c) computer readable program code calculating a new ID value based upon node ID value(s) identified in (b), said calculated value greater than ID values of nodes to the left of said insertion point and less than ID values of nodes to the right of said insertion point, said new ID value based upon a low/high key value, said high key value representing a highest encodable value and said low key value representing a lowest encodable value; and

(d) computer readable program code encoding said calculated new ID value and updating said computer storage with said encoded value, wherein order, node ID values, and relationships between parent, child, and siblings in said hierarchical structure of nodes stored in said storage remain unchanged with said insertion of new node.

32. (Original) An article of manufacture as per claim 31, wherein said new ID value is calculated via any of the following steps: concatenating said left node ID value with one or more high key values and a positive value or concatenating said left node ID value with one or more zeros and a positive value.

33. (Original) An article of manufacture as per claim 31, wherein said ID values are encoded and are byte comparable.

34. (Original) An article of manufacture as per claim 31, wherein said nodes are associated with a mark-up language based document.

35. (Original) An article of manufacture as per claim 34, wherein said mark-up based language is XML.

36. (Original) An article of manufacture as per claim 31, wherein said medium works in conjunction with a relational database.

37. (Previously Presented) An article of manufacture as per claim 31, wherein said encoding is binary encoding and said highest encodable value is 1111 and said lowest encodable value is 0000.

38. (Original) An article of manufacture as per claim 31, wherein a digit in said calculated ID value has a negative value.

39. (Previously Presented) A computer-based method for updating a computer-stored

hierarchical structure of nodes without renumbering existing node ID values associated with said hierarchical structure, said nodes of said hierarchical structure stored as binary encoded values in a computer storage, said method comprising the steps of:

(a) receiving an instruction to insert a new node at an insertion point in said computer-stored hierarchical structure;

(b) identifying one of, or a combination of the following: a left node ID value closest to the left of said insertion point or a closest right node ID value closest to the right of said insertion point;

(c) calculating a new ID value for node to be inserted based upon a low key value 0 or a high key value x, said high key value representing a highest binary encodable value and said low key value representing a lowest binary encodable value, said calculation performed via one of the follows ways: concatenating said left node ID value with one or more high key values and a positive value or concatenating said left node ID value with one or more low key values and a positive value; and

(d) encoding, via binary encoding, said calculated new ID value and updating said computer storage with said binary encoded value, wherein order, node ID values, and relationships between parent, child, and siblings in said hierarchical structure of nodes stored in said storage remain unchanged with said insertion of new node.

40. (Original) A computer-based method as per claim 39, wherein a digit in said calculated ID value has a negative value.

41. (Original) A computer-based method as per claim 39, wherein said ID values are

encoded and are byte comparable.

42. (Original) A computer-based method as per claim 39, wherein said nodes are associated with a mark-up language based document.

43. (Original) A computer-based method as per claim 42, wherein said mark-up based language is XML.

44. (Original) A computer-based method as per claim 39, wherein said method is implemented in conjunction with a relational database.

Evidence Appendix

None

Related Proceedings Appendix

None